CSIE 1000 Introduction to Computers **National Taiwan University**
Fall 2016 **Department of CSIE**

Homework 1

October 25, 2016 Due date: November 8, 2016

1. (12%) What are the 8-bit 2's complement representations of the following decimal numbers? Please give both their binary and hexadecimal representations.
a. 95
b. -6
c. -29

2. (8%) Prove that (a) $\overline{A}BC + ABC + B\overline{C} = B$ and (b) $A + \overline{A}B = A + B$ using Boolean algebra.

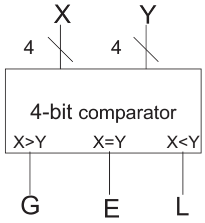
3. (8%) A 4-to-2 encoder has four inputs A$_3$, A$_2$, A$_1$, A$_0$ and two outputs Z$_1$, Z$_0$. Only one of the four inputs can be 1 at a time. Assume that A$_i$ is on, the output (Z$_1$, Z$_0$) will correspond to the binary representation of i. For example, when the input A$_2$=1, (Z$_1$, Z$_0$)=(1,0) because $10_2 = 2_{10}$. Create the truth table for the 4-to-2 encoder and implement it with logic gates.

4. (8%) Design and implement a 4-bit right shifter with sign extension. The inputs are a 2-bit shift amount S_1S_0 and a 4-bit number $X_3X_2X_1X_0$, where X_3 is the MSB. The output is a 4-bit number $Z_3Z_2Z_1Z_0$.

5. (10%) In Hack ALU, the following configurations of inputs are used for $x - 1$ and $x - y$. Explain why they work.

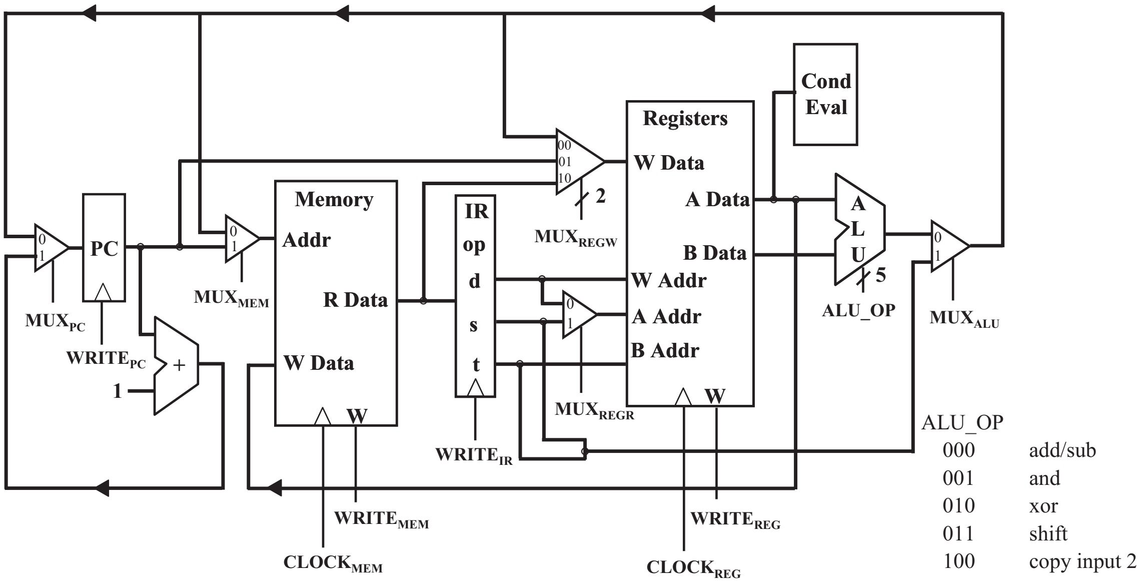
zx	nx	zy	ny	f	no	out
0	0	1	1	1	0	$x - 1$
0	1	0	0	1	1	$x - y$

6. (16%) As shown in the following diagram, design a 4-bit comparator which has two 4-bit unsigned integer inputs, $X_3X_2X_1X_0$ and $Y_3Y_2Y_1Y_0$, and a 3-bit output for the conditions of $X > Y$, $X = Y$ and $X < Y$, respectively. *Hint: design a 1-bit comparator first.*



7. (16%) Design a binary multiplier that multiplies two 3-bit unsigned integers, $X = X_2X_1X_0$ and $Y = Y_2Y_1Y_0$, with a 6-bit output $Z = Z_5Z_4 \dots Z_0$ where $Z = X \times Y$; X_0, Y_0 and Z_0 are LSBs. You may use the notation $X[n..m]$ to identify a portion of wires. For example, X[2..1] means the set of wires, X_2X_1.

8. (12%) Refer to the TOY architecture (Figure 1; note that the numbering could be different from the lecture), please specify the operations of MUX_{PC}, MUX_{MEM}, MUX_{REGR}, MUX_{ALU}, MUX_{REGW}, $WRITE_{REG}$, $WRITE_{MEM}$ and ALU_OP during the execution stage for the instructions "xor", "store indirect" and "branch if zero". As an example, for "jump and link", they would be $MUX_{PC} = 0, MUX_{MEM} = *, MUX_{REGR} = *, MUX_{ALU} = 1, MUX_{REGW} = 01, WRITE_{REG} = 1, WRITE_{MEM} = 0, ALU\_OP = *$. (For ALU_{OP}, you only need to specify 3-bit $ALU_{control}$ as specified in Figure 1.)



ALU_OP	
000	add/sub
001	and
010	xor
011	shift
100	copy input 2

Figure 1: TOY architecture.

9. (10%) Assume that the program starts at 10 and the following data appear on standard input.

```
1112 1112
```

What is output of the following TOY program?

```
10:  7100
11:  8FFF
12:  9F15
13:  82FF
14:  1112
15:  C016
16:  91FF
17:  0000
```